2. Strings

Na linguagem C uma string é um vetor de caracteres. Porém, obrigatoriamente um dos caracteres do vetor deve ser o caractere nulo, ou seja, o '\0'. O caractere nulo sucede o último caractere válido da string em questão. Para declarar uma string, podemos usar o seguinte formato geral:

char nome_da_string [tamanho];

Vetores e Strings

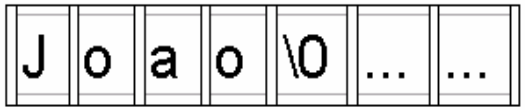
Exemplo:

char n [7];

Se inicializarmos a string de 7 posições declarada acima colocando nela a palavra Joao nela, da seguinte forma:

char n [7]="Joao";

Teremos:



Vetores e Strings

Formas de inicialização:

char n [7]="Joao"; ou char n []="Joao";

ou

char n []={'J','o','a','o','\0'};

ou

char n [7];
n [0]='J';
n [1]='o';
n [2]='a';
n [3]='o';
n [4]='\0';

Observação:

...
char str[10] = "Joao";
...
~~**str = "maria";**~~

Vetores e Strings

Como ler uma *string* através da entrada padrão?

Podemos utilizar a função *scanf* com o código %s.

Qual é o parâmetro que deve ser fornecido?

Devemos fornecer o endereço de memória onde a *string* deve ser armazenada, ou melhor, devemos fornecer o endereço de onde deve-se iniciar o armazenamento da *string*. Esta informação é obtida através do identificador do vetor de caracteres que conterá a *string*. Exemplo:

```
...
char n [20];
...
scanf (“%s”,n);     ou     scanf (“%s”,&n[0]);
```

Vetores e Strings

Como escrever uma *string* na saída padrão?

Podemos utilizar a função *printf* com o código %s.

Qual é o parâmetro que deve ser fornecido?

Devemos fornecer o endereço de memória onde a *string* está armazenada, ou melhor, devemos fornecer o endereço de memória onde encontra-se armazenado o primeiro caractere da *string*. Esta informação é obtida através do identificador do vetor de caracteres que contém a *string*. Exemplo:

```
...
char n [20];
...
printf (“%s”,n);     ou     printf (“%s”,&n[0]);
```

2. Strings (continuação)

Exercício: Construa um programa que leia através da entrada padrão uma string e retorne na saída padrão o numero de caracteres que a mesma possui.

Vetores e Strings

2. Strings (continuação)

Exercício: Construa um programa que declare duas strings, string1 e string2 respectivamente, ambas com capacidade para armazenar 20 caracteres válidos, o programa deve ler através da entrada padrão uma string e colocá-la na string1, depois o programa deve atribuir o conteúdo da string1 para a string2 e imprimi-la na saída padrão.

Vetores e Strings

3. Vetores de Strings

Se fizermos um vetor de strings estaremos construindo um vetor de vetores. Esta estrutura é uma matriz bidimensional de **char**'s. Podemos ver a forma geral de uma vetor de strings como sendo:

char nome_da_variável [num_de_strings][compr_das_strings];

3. Vetores de Strings (continuação)

Aí surge a pergunta: como acessar uma string individual?

Fácil. É só usar apenas o primeiro índice. Então, para acessar uma determinada string faça:

nome_da_variável [índice]

Exemplo: O programa a seguir declara um vetor de string’s, o inicializa com string’s fornecidas através da entrada padrão e no final de seu processamento o retorna na saída padrão.

```
#include <stdio.h>
main ()
{
   char strings [5][100];
   int count;
   for (count=0;count<5;count++)
   {
      printf ("\n\nDigite uma string: ");
      scanf (“%s”,strings[count]);
   }
   printf ("\n\nAs strings que voce digitou foram:\n");
   for (count=0;count<5;count++)
      printf ("%s\n",strings[count]);
}
```

Vetores e Strings

3. Vetores de Strings (continuação)

Exercício:

Construa um programa que, com base no exemplo anterior, além de ler as 5 string's do vetor de strings leia mais uma string, a qual ele verificará se pertence ao vetor, caso esta pertença ele retornará a posição da string no vetor, caso contrario ele retornará uma mensagem indicando que ela não se encontra no vetor.